AMENDMENTS TO THE SPECIFICATION

Please replace the paragraph at page 6, from lines 19 – 21, with the following amended paragraph:

Figs. ~~3 and 4~~ 3A, 3B, 4A and 4B are flow diagrams detailing a process in accordance with an embodiment of the present invention, and detail elements of the respective processes in accordance with standard programming conventions;

Please replace the paragraph at page 6, line 29 to page 7, line 7, with the following amended paragraph:

Fig. 1 shows an exemplary data structure representing a Direct Cyclic Graph (DCG), that will be used in describing an exemplary packet classification process in accordance with the invention. This Direct Cyclic Graph (DCG) is used to store all of the information related to a classification process, such as that shown in Figs. ~~3 and 4~~ 3A and 3B and Figs. 4A and 4B, and detailed below. The DCG has interconnected edges and nodes. Each edge of a graph has a special pattern (detailed below) associated with it. Each node has a unique number assigned to it, which is used to represent a path in a DCG. When a pattern matches a packet being classified, it means that this DCG can be traversed to the next connecting node.

Please replace the paragraph at page 9, from lines 11 – 13, with the following amended paragraph (please note that the amendment to this paragraph is to indent the first word of the amended paragraph, and accordingly, which can not be expressed by underline or strikethrough):

An optional ***expire*** flag indicates the time period (in milliseconds) after which the loaded cell value must be discarded (expired). If not specified, a default expiration value of 60000 should be used.

Please replace the paragraph at page 15, lines 4-11, with the following amended paragraph:

Turning also to ~~Fig. 3~~ Figs. 3A and 3B, there is detailed a process, in the form of a flow diagram, in accordance with an embodiment of the present invention. The process of ~~Fig. 3~~ Figs. 3A and 3B will be described with respect to the DCG of Fig. 1, with the numbers of the nodes corresponding thereto. This process is typically passive, as it typically does not modify packet data. The process results in incoming packets being classified in accordance with their individual characteristics as well as those of its data flow. The classification process as described herein is such that it modifies the DCG data dynamically and continuously (on-the-fly).

Please replace the paragraph at page 15, lines 23-27, with the following amended paragraph:

Alternately, if the list is not empty, as there are outgoing edges, the pattern is taken in accordance with its order on the list at block 110. The pattern includes seven expressions, known as **expr0**, ..., **expr6**. The first expression (expr0) is then calculated at block 112. The calculation process of block 112 will now be described by reference to ~~Fig.~~4 Figs. 4A and 4B.

Please replace the paragraph at page 17, lines 16-21, with the following amended paragraph:

expr0 has now been calculated, in accordance with the flow diagram of ~~Fig. 4~~ Figs. 4A and 4B, and if false, the process moves to block 114 (Fig. 3A), where packet state is looked up in a state set (as shown on Fig. 2). The number of the set used for this lookup is specified by the

pattern's variable **set** as described above. At block 116, if the lookup was successful, i.e., a value for the packet state was located, the process moves to block 130. Alternately, if the lookup was unsuccessful, the process returns to block 110.

Please replace the paragraph at page 17, lines 22-25, with the following amended paragraph:

Returning to block 112, if expr0 has been calculated to be true, the process moves to block 120, where all remaining expressions of the pattern (if any) are calculated according to the process of ~~Fig. 4~~ Figs. 4A and 4B. Regardless of any other pattern expressions and their results, the process moves to block 130.

Please replace the paragraph at page 18, lines 20-23, with the following amended paragraph:

Turning now to Fig. 5, there is shown a Classification Engine (CE) 300 on which the processes of embodiments of the invention, for example, as shown in ~~Fig. 3 and 4~~ Figs. 3A and 3B and Figs. 4A and 4B, and described above, can be performed. Typical operating systems employing this engine 300 utilize kernel space 302, separate from user space 304.

Please replace the paragraph at page 40, lines 20-24, with the following amended paragraph:

This example is directed to an implementation of the process detailed in Figs. ~~3 and 4~~ 3A and 3B and Figs. 4A and 4B. This example details memory requirements and execution time of the aforementioned process. For this particular implementation, to ensure high performance of

the algorithm and to maintain a minimal memory footprint, the following rules were applied.

These rules were as follows: